Diskrete Logarithmen auf elliptischen Kurven

Kryptographie mit elliptischen Kurven beruht meist auf dem diskreten Logarithmenproblem: Für reelle Zahlen ist der Aufwand zur Berechnung des Logarithmus vergleichbar mit dem zur Berechnung der Exponentialfunktion; dies hängt wesentlich damit zusammen, daß beides streng monoton wachsende stetige Funktionen sind. Das ändert sich dramatisch, wenn man die reellen Zahlen durch einen endlichen Körper ersetzt und sich dann natürlich auf ganze Zahlen als Exponenten beschränken muß. Betrachten wir etwa die Funktion

$$\begin{cases} \mathbb{Z}/18 \to \mathbb{F}_{19}^{\times} \\ \quad x \mapsto 10^x \end{cases},$$

so zeigt die folgende Wertetabelle ein völlig chaotisches Verhalten:

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
10^x	1	10	5	12	6	3	11	15	17	16	9	14	7	13	16	8	4	2

(Da nach dem kleinen Satz von FERMAT $a^{18} = 1$ ist für alle $a \in \mathbb{F}_{19}^{\times}$, können wir uns auf den Definitionsbereich $\mathbb{Z}/18$ beschränken.)

Auf der Menge der ganzen Zahlen ist die Funktion $x \mapsto 10^x$ zwar immer noch streng monoton wachsend, aber dadurch, daß wir die Ergebnisse modulo 19 betrachten, wird diese Monotonie sehr effizient zerstört. Es ist daher nicht mehr möglich, einfach anhand der Größe von 10^x auf den Exponenten x zu schließen; wenn wir wissen wollen, für welches x beispielsweise $10^x \equiv 4 \bmod 19$ ist, bleibt uns zumindest als erster Ansatz nichts anderes übrig, als solange Potenzen von zehn modulo 19 zu berechnen, bis wir für $x = 16$ das Ergebnis vier erhalten. $10^{16} \bmod 19$ dagegen können wir leicht direkt durch viermaliges Quadrieren berechnen:

$$10^2 = 100 \equiv 5 \bmod 19$$
$$10^4 \equiv 5^2 = 25 \equiv 6 \bmod 19$$
$$10^8 \equiv 6^2 = 36 \equiv 17 \equiv -2 \bmod 19$$
$$10^{16} \equiv (-2)^2 = 4 \bmod 19$$

Bei so kleinen Zahlen ist die Bestimmung des Exponenten natürlich kein großes Problem; falls wir aber $p = 19$ durch eine hundertstellige Primzahl ersetzen, können wir für eine feste Basis a die Zahl $a^x \mod p$ zwar immer noch mit relativ geringem Aufwand berechnen: Wie wir wissen, brauchen wir etwa $\log_2 x$ Quadrierungen und im Mittel ungefähr halb so viele weitere Multiplikationen modulo p, was für $p \approx 10^{100}$ ungefähr fünfhundert Rechenoperationen sind. Wenn wir aber bestimmen wollen, für welches x die Potenz $a^x \mod p$ einen vorgegebenen Wert y hat, brauchen wir bei dieser naiven Vorgehensweise durch Probieren im Mittel etwa $p/2$ Versuche, was deutlich über der Schranke von 2^{120} oder auch 2^{128} liegt, von der man annimmt, daß heutzutage niemand so viele Rechenoperationen durchführen kann.

Nun gibt es natürlich bessere Methoden zur Berechnung des Exponenten als das systematische Probieren; bei allen in der offenen Literatur bekannten Methoden ist aber der Aufwand zur Berechnung des Exponenten deutlich größer als der zur Berechnung der Potenz. Nach den Empfehlungen des Bundesamts für Sicherheit in der Informationstechnik sollten Kryptoverfahren auf der Basis diskreter Logarithmen derzeit sicher sein, wenn p mindestens 2000 Bit lang ist; ab Ende 2022 allerdings sollten es schon 3000 Bit sein, und das zuständige EU-Gremium empfiehlt diese Länge bereits heute.

Die beste derzeit bekannte Methode zur Bestimmung einer Zahl x mit $a^x \equiv y \mod p$ ist der sogenannte Indexkalkül, ein relativ komplizierter Algorithmus, der wesentlich darauf beruht, daß nicht nur x eine ganze Zahl ist, sondern daß auch $a^x \mod p$ in natürlicher Weise mit einer solchen identifiziert werden kann.

Hier kommen nun elliptische Kurven ins Spiel: Auch dort haben wir eine Art diskrete Exponentialfunktion: Ist nämlich E eine elliptische Kurve und $P \in E$ ein Punkt der Ordnung q, so haben wir die Abbildung

$$\begin{cases} \mathbb{Z}/q \to E \\ \quad x \mapsto xP \end{cases},$$

und das diskrete Logarithmenproblem besteht darin, für einen Punkt Q aus dem Bild die Zahl x zwischen 0 und $q - 1$ zu bestimmen, für die

$xP = Q$ ist. Da Punkte einer elliptischen Kurve nicht in natürlicher Weise als ganze Zahlen betrachtet werden können, ist hier eine Methode nach Art des Indexkalküls nicht anwendbar, so daß man sich auf weniger effiziente Ansätze beschränken muß. Auf diese Weise können die Parameter deutlich kleiner gewählt werden; das Bundesamt für Sicherheit in der Informationstechnik empfiehlt für q eine Primzahl mit 256 Bit.

Der Grund für die Primzahlempfehlung liegt darin, daß man das diskrete Logarithmenproblem in einer zyklischen Gruppe der Ordnung n über den chinesischen Restesatz zurückführen kann auf diskrete Logarithmen in Untergruppen, deren Ordnungen die Primzahlpotenzen sind, die n teilen, und nach einem Algorithmus von POHLIG und HELLMAN läßt sich die Berechnung in einer Gruppe von Primzahlpotenzordnung reduzieren auf Berechnungen in Gruppen, deren Ordnung die Primzahl selbst ist. Maßgeblich für die Sicherheit ist daher nicht die Ordnung n der Gruppe, sondern der größte Primteiler von n.

Eine Gruppenordnung von etwa 2^{256} bedeutet, wie wir noch sehen werden, daß die elliptische Kurve über einem Körper $\mathbb{F}_p$ definiert sein sollte, für den p auch ungefähr 256 Bit hat, wir müssen also nur mit Zahlen dieser Größenordnung rechnen statt, wie bei klassischen Verfahren auf der Grundlage diskreter Logarithmen, mit solchen einer Länge von 2000 oder gar 3000 Bit. Obwohl der Aufwand für die Addition zweier Punkte auf einer elliptischen Kurve deutlich höher ist als der für eine modulare Multiplikation, sorgt dieser gewaltige Größenunterschied dafür, daß Verfahren auf der Grundlage elliptischer Kurven bei gleicher Sicherheit schneller sind als solche in endlichen Körpern.

Das älteste Verahren auf der Grundlage diskreter Logarithmen ist der Schlüsselaustausch nach DIFFIE-HELLMAN. Zwei Personen wollen für ihre Kommunukation einen Schlüssel für ein klassisches Kryptoverfahren vereinbaren, haben aber keine Möglichkeit, sich dazu zu treffen, sondern können nur über eine unsichere Leitung miteinander kommunizieren.

Beim klassischen DIFFIE-HELLMAN-Verfahren einigen sich die beiden Teilnehmer zunächst (über die unsichere Leitung) auf eine Primzahl p und eine natürliche Zahl a derart, daß die Potenzfunktion $x \mapsto a^x$

möglichst viele Werte annimmt (und deren Anzahl einen großen Primteiler hat). Als nächstes wählt Teilnehmer A eine Zufallszahl $x < p$ und B entsprechend ein $y < p$. A schickt $u = a^x \bmod p$ an B und erhält dafür $y = a^y \bmod p$ von diesem. Sodann berechnet A die Zahl

$$v^x \bmod p = \left(a^y\right)^x \bmod p = a^{xy} \bmod p$$

und B entsprechend

$$u^y \bmod p = \left(a^x\right)^y \bmod p = a^{xy} \bmod p\,;$$

beide haben also auf verschiedene Weise dieselbe Zahl berechnet, die sie zum Beispiel verwenden können, um daraus einen Schlüssel für ein symmetrisches Kryptosystem zu bestimmen. Verfahren dazu gibt es mehr als genug: Sie könnten etwa die letzten oder sonst irgendwelche Bits dieser Zahl verwenden, aber auch einen irgendwie definierten Hashwert.

Ein Gegner, der den Datenaustausch abgehört hat, kennt die Zahlen p, a, u und v; er kann also problemlos alle möglichen Zahlen modulo p der Art $a^{\alpha x+\beta y} = u^\alpha \cdot v^\beta$ berechnen. Es fällt aber schwer, sich eine Art und Weise vorzustellen, wie er $a^{xy} \bmod p$ finden kann, ohne den diskreten Logarithmus von u oder v zu berechnen. (Bewiesen ist hier, wie in der Kryptologie üblich, natürlich nichts.)

Die Übertragung des DIFFIE-HELLMAN-Verfahrens auf elliptische Kurven ist offensichtlich: A und B einigen sich über die unsichere Leitung auf eine elliptische Kurve E über einem endlichen Körper $\mathbb{F}_p$ und einen Punkt $P \in E$ mit möglichst großer Ordnung q, die am besten eine Primzahl sein sollte. Dann wählt A eine Zufallszahl $x < q$, berechnet (nach dem Algorithmus von MONTGOMERY) den Punkt $U = xP$, und schickt ihn an B. Dieser wählt ein $y < q$ und schickt $V = yP$ an A. Nun können beide den Punkt $xyP = xV = yU$ berechnen, aus dessen Koordinaten sie in irgendeiner Weise (die über die unsichere Leitung vereinbart werden kann) den gewünschten Schlüssel bestimmen.

Tatsächlich wird dieses Verfahren nur selten angewandt, denn es läßt sich durch eine sogenannte *man in the middle attack* angreifen: Da der Gegner die Wahl der Mittel hat, muß er nicht notwendigerweise die mathematische Seite des Verfahrens angreifen, sondern kann sich eine beliebige Schwachstelle aussuchen.

Nehmen wir etwa an, der Angreifer habe eine gewisse Kontrolle über das Netz, in dem der Datenaustausch stattfindet – beispielsweise, weil er Systemverwalter eines für die betreffende Verbindung unbedingt notwendigen Knotenrechners ist. Dann kann er als *man in the middle* alle Datenpakete zwischen A und B abfangen und durch selbstfabrizierte eigene Pakete ersetzen

Damit kann er sich gegenüber A als B auszugeben und umgekehrt: Alles, was A an B zu schicken glaubt, geht tatsächlich an den Gegner G, und alles was B von A zu erhalten glaubt, kommt tatsächlich von G. In Gegenrichtung ist es natürlich genauso.

Im einzelnen läuft der Angriff folgendermaßen ab:

Falls die Zahlen a und p oder die Kurve E und der Punkt P nicht ohnehin Konstanten eines Verbunds sind, dem A und B angehören, läßt G die Kommunikation, die zu deren Vereinbarung führt, ungehindert zu: In diesem Stadium beschränkt er sich auf reines Abhören.

Als nächstes wählen A und B ihre Zufallszahlen x und y; gleichzeitig wählt G eine Zufallszahl $z < p$ oder vielleicht auch zwei verschiedene solche Zahlen z_A und z_B für die beiden Teilnehmer.

Wenn A die Zahl $u = a^x \bmod p$ oder den Punkt $U = xP$ an B schickt, fängt G diese Nachricht ab und ersetzt sie durch $w_B = a^{z_B} \bmod p$ *bzw.* $W_B = z_B P$; entsprechend fängt er Bs Nachricht ab und schickt stattdessen $w_A = a^{z_A}$ *bzw.* $W_A = z_A P$ an A. Dies führt dazu, daß am Ende A und G einen gemeinsamen Schlüssel s_A haben und B und G einen gemeinsamen Schlüssel s_B. Sowohl A als auch B glauben, der ihnen bekannte Schlüssel s_A *bzw.* s_B sei aus $a^{xy} \bmod p$ abgeleitet und senden nun damit verschlüsselte Nachrichten an ihren Partner. Diese Nachrichten fängt G ab, entschlüsselt sie mit dem Schlüssel, den er mit dem Absender gemeinsam hat, und verschlüsselt sie anschließend, gegebenenfalls nach einer seinen Interessen entsprechenden Modifikation, mit dem Schlüssel, den er mit dem Empfänger gemeinsam hat. Auf diese Weise hat er die gesamte Konversation unter Kontrolle, ohne daß A und B etwas merken.

Die Möglichkeit für diese Attacke kommt natürlich daher, daß sich A und B nicht sicher sein können, den jeweils anderen am anderen Ende

der Leitung zu haben. Die kryptographisch einwandfreie Modifikation, die das Verfahren gegen diese Art von Angriff sicher macht, bestünde beispielsweise darin, daß A und B ihre Nachrichten x und y vor dem Versenden digital unterschreiben – aber dann verschwindet auch wieder der Vorteil, daß sie ohne Kenntnis irgendeines Schlüssels miteinander kommunizieren können: Zur Verifikation einer Unterschrift braucht man schließlich den öffentlichen Schlüssel des Unterschreibenden. (Verfahren für digitale Unterschriften werden wir bald kennenlernen, denn sie sind die Hauptanwendung der Kryptographie mit elliptischen Kurven.)

Falls sich A und B hinreichend gut kennen, um die Stimme des jeweils anderen am Telephon einigermaßen sicher zu erkennen, können sie diese Art von Attacke auch dadurch erschweren, daß sie nach dem Austausch von u und v per Telephon über diese Zahlen (z.B. die 317. bis 320. Ziffer) und gegebenenfalls auch noch über Schwänke aus ihrer gemeinsamen Jugendzeit reden; dann müßte der Angreifer zusätzlich noch ein begabter, kundiger und reaktionsschneller Stimmenimitator sein, der auch die Telephonverbindung als *man in the middle* so angreifen kann, daß weder A noch B etwas merkt. Bei Videokonferenzen könnte man auch die Zahlen langsam über den Bildschirm des jeweils anderen laufen lassen. Die volle Sicherheit einer Schlüsselvereinbarung mit digital unterschriebenen Nachrichten wird aber nicht erreicht, und da oft zumindest einer der Teilnehmer ein Unternehmen ist, das sich einen zertifizierten öffentlichen Schlüssel zur Verifikation seiner Unterschrift leisten kann, werden Schlüssel für symmetrische Kryptoverfahren in der Praxis sehr viel häufiger so vereinbart als via DIFFIE-HELLMAN.

Im *electronic banking* wird die Idee eines zweiten Kommunikationskanals trotzdem häufig angewandt, hier im allgemeinen dadurch, daß ein Teil des Protokolls via SMS abläuft. Auch die können zwar selbstverständlich manipuliert werden, aber der Aufwand eines Angreifers steigt ganz beträchtlich, wenn er *gleichzeitig* zwei verschiedene Verbindungen manipulieren muß: Die meisten *phishing*-Attacken arbeiten schließlich mit gefälschten Webseiten im Ausland, und selbst im Inland ist es nicht so einfach, Mobilfunkverbindungen in einem größeren Gebiet zu überwachen und zu manipulieren.

In der nächsten Vorlesung werden wir sehen, wie man die Idee des

Schlüsselaustauschs nach DIFFIE-HELLMAN so modifizieren kann, daß sowohl Verschlüsselungsverfahren mit öffentlichen Schlüsseln als auch Verfahren für elektronische Unterschriften entstehen. Zum Abschluß der heutigen Vorlesung möchte ich stattdessen ein Verfahren zur Berechnung diskreter Logarithmen vorstellen, das deutlich schneller ist als Probieren, und das sowohl für den klassischen Fall als auch für elliptische Kurven funktioniert.

Der *baby step – giant step* Algorithmus des amerikanischen Mathematikers DANIEL SHANKS (1917–1996) reduziert auf Kosten von mehr Speicherplatz die Rechenzeit recht deutlich gegenüber reinem Probieren: Ist q die Ordnung des Punktes P (oder von $a \in \mathbb{F}_p^\times$ im klassischen Fall), so ist der Aufwand nicht mehr proportional zu q, sondern nur noch zu $\sqrt{q} \cdot \log_2 q$.

SHANKS stellte sein Verfahren natürlich für den klassischen Fall vor; wir betrachten gleich die (offensichtliche) Modifikation für elliptische Kurven. Er wählt eine natürliche Zahl m die ungefähr gleich $\sqrt{q} \cdot \log_2 q$ ist; falls man a nicht so genau kennt, kann zwar die Effizienz des Verfahrens unter einer schlechten Wahl von m geringfügig leiden, aber solange die Größenordnungen einigermaßen stimmen, ist das nicht so dramatisch. Wichtig ist nur, daß $m > \sqrt{q}$ ist, aber nicht dramatisch größer.

Danach berechnet er die sämtlichen Vielfachen iP von P mit $i \leq m$; das sind m sogenannte *baby steps*.

Bei den dann folgenden *giant steps* berechnet er, um den diskreten Logarithmus eines Punkts Q aus der von P erzeugten Untergruppe von E zu erhalten, die Punkte $Q - mjP$ für $j = 1, 2, \ldots$ und vergleicht sie mit den Punkten iP aus dem ersten Teil. Ein solcher Vergleich kann etwa über eine binäre Suche oder eine *hash*-Tabelle implementiert werden und hat einen Aufwand proportional $\log_2 m$.

Sobald ein Punkt $Q - mjP$ gefunden ist, der mit einem der in den *baby steps* berechneten Punkte iP übereinstimmt, gilt $Q - mjP = iP$, also $Q = (mj + i)P$, und der diskrete Logarithmus ist gefunden.